BPFLH: Byzantine-Robust Privacy-Preserving Federated Learning for Heterogeneous Data

Guofu Zhu, Wenting Shen, Zhiquan Liu, *Senior Member*, *IEEE*, Jing Qin, and Jixin Ma

***Abstract*—Byzantine-robust federated learning (FL) aims to obtain an accurate global model even with potentially Byzantine users. However, most existing schemes rely on measuring the overall differences between the entire gradient vectors of different users, which fail to effectively distinguish malicious gradients from benign ones caused by data heterogeneity under non-IID settings, thereby compromising model performance. To tackle this challenge, we propose BPFLH, a novel Byzantine-robust privacy-preserving FL framework for heterogeneous data. BPFLH is the first to introduce Bray–Curtis dissimilarity into FL, capturing the element-wise differences among gradients from different users. This method reduces the risk of misclassifying benign gradients as malicious and enhance the model's robustness against Byzantine attacks in non-IID data environments. Furthermore, BPFLH leverages CKKS homomorphic encryption to protect local gradients, enabling secure aggregation and Byzantine user detection without compromising privacy. Extensive experiments on real-world datasets under various attack scenarios and data distributions demonstrate that BPFLH exhibits strong robustness against Byzantine attacks while preserving privacy and maintaining superior accuracy compared to existing Byzantine-robust FL methods, particularly in non-IID environments.**

***Index Terms*—Federated learning, Byzantine-robustness, privacy-preserving, heterogeneous data.**

I. Introduction

With the rapid development of the Internet, artificial intelligence (AI) technologies, particularly machine learning, have flourished in recent years and found extensive applications across various domains. However, vast amounts of data are distributed across numerous devices, and these data often contain sensitive information of the users. Growing privacy awareness and regulatory frameworks such as the General Data Protection Regulation (GDPR) prohibit the direct exchange of personal data, which results in the problem of data silos [1]. Federated Learning (FL), a distributed machine learning paradigm introduced by Google [2], provides an effective solution to this issue. In FL, data remain on local devices, and only the local gradients are shared with a central server, which then aggregates these local gradients to construct a global model. In this manner, FL alleviates the data silo problem while safeguarding user privacy. Moreover, FL enables multiple users to collaboratively train a model with stronger generalization capabilities, thereby attracting widespread interest.

Despite these advantages, FL is highly susceptible to Byzantine attacks, which threaten the reliability of the global model [3]. In such attacks, some users may be compromised by adversaries and become malicious (Byzantine) users [4]. These users may conduct data poisoning attacks, where manipulated local datasets lead to corrupted gradients [5], or model poisoning attacks, where gradients are directly tampered with before being uploaded [6]. When these malicious gradients are aggregated by the central server, they can significantly degrade the global model or even cause it to diverge [7], [8]. To defend against such attacks, a variety of Byzantine-robust FL schemes [9], [10], [11] have been proposed. These defenses typically detect malicious gradients using global dissimilarity-based metrics, such as Euclidean distance [12], [13], [14] or cosine similarity [15], [16], [17], which measure the overall differences between the entire gradient vectors of Byzantine and benign users. Once identified, the malicious gradients are either removed or assigned reduced weights during aggregation, thereby mitigating or eliminating their impact on the global model [15], [16].

However, these defense mechanisms, which rely on global dissimilarity-based metrics, face two key challenges. First, adversaries can evade detection by introducing subtle perturbations to their gradients, making malicious updates difficult to distinguish from benign ones [18]. Second, due to data heterogeneity in FL, local datasets across users are typically non-independent and identically distributed (non-IID), as their data distributions differ significantly. For example, in a medical federated learning scenario, hospitals in different field may receive patient populations with distinct health conditions and disease distributions. A hospital specializing in cardiovascular diseases is likely to have a higher proportion of cardiovascular disease-related data in its patient data center, whereas a hospital focusing on orthopedics will have more data related on orthopedics diseases. In a data environment characterized by heterogeneity, differences in users' local data distributions can lead to significant variations in the element-wise values of their gradients. These variations do not necessarily indicate Byzantine attacks but may simply reflect the natural differences in data distributions. As a result, these defense methods may fail to effectively distinguish between deviations caused

This research is supported by National Natural Science Foundation of China (U24A20244), Shandong Provincial Natural Science Foundation, China (ZR2025MS1006) and Youth Innovation Technology Project of Higher School in Shandong Province (2023KJ365).

G. Zhu and W. Shen are with the College of Computer Science and Technology, Qingdao University, Qingdao 266071, China. E-mail: zhuguofu05@163.com, shenwentingmath@163.com. (*Corresponding author: W. Shen*)

Z. Liu is with the College of Cyber Security, Jinan University, Guangzhou 510632, China. E-mail: zqliu@vip.qq.com.

J. Qin is with the School of Mathematics, Shandong University, Jinan, 250100, China. E-mail:qinjing@sdu.edu.cn.

J. Ma is with is the director and academic-reader of Computing and Mathematical Sciences Department at University of Greenwich, the United Kingdom. E-mail:j.ma@greenwich.ac.uk.

by data heterogeneity and those resulting from Byzantine attacks, potentially misclassify benign gradients as malicious and degrading the global model performance.

Besides robustness concerns, privacy risks also arise in FL. Recent studies [19], [20] have revealed that directly uploading local gradients can still expose sensitive information, as adversaries may exploit these gradients to conduct gradient inference attacks and recover users' private data. To mitigate this risk, many privacy-preserving FL schemes employ cryptographic techniques to encrypt local gradients before sharing, allowing them to be aggregated without exposing sensitive information. Among these techniques, Homomorphic Encryption (HE) has been widely adopted because it enables computations to be performed directly on encrypted data, thereby ensuring privacy throughout the entire training process [21], [22].

However, while privacy-preserving methods effectively protect sensitive data, most of them do not address Byzantine attacks. Existing privacy-preserving approaches and Byzantine-robust defense mechanisms in federated learning differ substantially in their design goals: the former focuses on safeguarding user privacy, whereas the latter aims to mitigate malicious updates by evaluating their dissimilarity with benign gradients. Therefore, developing algorithms that can simultaneously preserve privacy and enhance robustness, while accurately identifying malicious gradients, remains an important open challenge. This challenge becomes even more difficult under non-IID settings, where data heterogeneity exacerbates gradient divergence and makes malicious gradients harder to distinguish from benign ones.

To overcome these issues, we propose BPFLH, a novel Byzantine-robust and privacy-preserving FL scheme for heterogeneous data, which enhances Byzantine robustness in non-IID settings while protecting user privacy. The main contributions of BPFLH are summarized as follows:

1) We are the first to introduce a defense mechanism based on Bray–Curtis dissimilarity into FL. Unlike global dissimilarity-based metrics that measure overall differences between entire gradient vectors from different users, Bray–Curtis dissimilarity captures element-wise differences among these gradients. This fine-grained perspective makes it more resilient to the misleading effects of non-IID data, thus reducing the risk of misclassifying benign gradients as malicious and enhancing the model's robustness against Byzantine attacks in non-IID data environments.
2) By leveraging CKKS homomorphic encryption, BPFLH ensures that both the secure aggregation of benign users' local gradients and the identification of malicious gradients are performed without revealing any sensitive user information. In particular, the operations required for computing the Bray–Curtis dissimilarity can be executed directly over encrypted gradients, thereby enabling effective Byzantine user detection without decryption.
3) We evaluate BPFLH on real-world datasets under both IID and non-IID settings, considering various attack scenarios and different proportions of Byzantine users. The results demonstrate that BPFLH can effectively filter out malicious gradients, while ensuring privacy protection and consistently maintaining high model accuracy. Compared with existing Byzantine-robust federated learning schemes, BPFLH demonstrates superior performance under non-IID data settings.

Organization: The remainder of this paper is organized as follows. Section II provides a concise review of related work. Section III introduces the preliminaries used in BPFLH. The system model, threat model, and design goals of BPFLH are described in Section IV. Section V presents the detailed construction of BPFLH. The security proofs and experimental evaluations are discussed in Sections VI and VII, respectively. Finally, Section VIII concludes the paper.

II. Related Work

Federated learning is vulnerable to Byzantine attacks, where malicious users perform data or model poisoning by corrupting local datasets or gradients. These malicious gradients can degrade or even destabilize the global model when aggregated. To address this issue, a majority of Byzantine-robust federated learning schemes [23], [24], [25] have been developed. Blanchard et al. [13] introduced Krum, a method that calculates the Euclidean distances between the local gradients of users and selects the gradient with the smallest sum of Euclidean distances to other users' gradients as the standard gradient. This standard gradient is then used to update the global model, defending against Byzantine attacks. They also proposed Multi-Krum, an extension that enhances Byzantine defense by selecting multiple candidate gradients and performing an averaging aggregation. Yin et al. [9] proposed the Median and Trimmed-mean methods, which compute the median or trimmed average of the local gradients, excluding outliers to mitigate Byzantine attacks. FLTrust [15] calculates the cosine similarity between user gradients and the server model, dynamically assigning weights to gradients and performing weighted aggregation to protect against malicious updates. Dong et al. [18] introduced the CareFL scheme, incorporating the Shapley value to detect Byzantine users. Building on this, CareFL+ was developed to improve Byzantine user detection in large-scale federated learning by grouping users and applying CareFL both within and across groups. Yan et al. constructed the RECESS scheme [26], which actively queries each participating user for the cosine similarity and l_2 norm between the constructed aggregate gradient and the user's uploaded gradient. By analyzing these responses, it effectively detects malicious user, thereby providing proactive defense against attacks.

However, these Byzantine defense methods do not address privacy concerns. In traditional FL systems, local gradients are transmitted to a central server, which may inadvertently expose sensitive user data. To tackle this issue, privacy-preserving federated learning schemes [21], [27], have been developed, which leverages cryptographic techniques such as homomorphic encryption [28], [29] and secret sharing [12], [30] to protect user data. These techniques ensure that computations can be performed on encrypted data, maintaining privacy throughout the training process. However, these privacy-preserving schemes exacerbate the challenge of Byzantine defense. Encrypted gradients are not directly inspectable, complicating the detection of malicious gradients.

Recent studies [31], [32], [33] have focused on addressing both privacy and Byzantine resilience simultaneously. So et al. [27] constructed a privacy-preserving Byzantine defense scheme for federated learning, utilizing secret sharing to protect user privacy. It defends against Byzantine attacks by calculating the Euclidean distances between users' local gradients and selecting local gradients for aggregation based on these distances. Ma et al. proposed ShieldFL [16], which utilizes double-trapdoor homomorphic encryption to protect user privacy and identifies Byzantine users by computing the cosine similarity between local gradients. Zhang et al. designed the LSFL scheme [12], which splits local gradients and adds Gaussian noise to preserve user privacy. LSFL introduces an algorithm where two servers collaboratively compute the distance between users' local gradients and the average gradient. It defends against Byzantine attacks by selecting the k users with the smallest distances for aggregation. Tang et al. constructed the FLAD scheme [29], in which CKKS and random permutation techniques are employed to protect privacy. It trains a feature extraction model using the server data for feature extraction and distinguishes malicious gradients by comparing cosine similarity and Euclidean distance with standard features. ESFL [34] utilizes functional encryption for privacy protection while setting up two servers: verifier and server. The verifier calculates the transformed distribution of all encrypted local models and clusters the distribution, and the server detects the clustering results using cosine similarity to filter out malicious users. Zhou et al. designed a community-oriented secure federated learning framework (CoS-HFL) [17]. This framework employs differential privacy to safeguard user data, while simultaneously using cosine similarity and information entropy within and across communities to effectively detect Byzantine users.

However, these methods that rely on global dissimilarity-based metrics of local gradients to detect malicious gradients still face significant challenges in non-IID environments. The heterogeneity in local data often causes benign gradients to deviate significantly, leading to misclassification of these gradients as malicious, thus harming model performance. Therefore, it is of significant importance to design a Byzantine-robust privacy-preserving federated learning scheme that can accurately differentiate between malicious gradients and benign gradients deviating due to data heterogeneity in non-IID environments, while minimizing the impact on model performance and safeguarding user privacy.

III. Preliminaries

A. Notations

Table I provides a summary of the notations used in this paper, along with their corresponding descriptions.

TABLE I: Notations and descriptions

Notations	Descriptions
U_i	The i-th user in the FL
D_i	Private dataset of user U_i
ω^t	Global model at the t-th FL training round
g_i	Local gradient of user U_i
$\lvert g_i\rvert$	Absolute gradient of user U_i
G	The aggregated gradient
μ	Learning rate
$BC_{i,j}$	Bray-Curtis dissimilarity between gradients of users U_i and U_j
d	The dimensionality of the gradient
pk	Public key used for encryption
sk	Secret key used for decryption
evk	Evaluation key used in homomorphic operations
s	The scaling factor
$g_i[k]$	The k-th element in local gradient g_i of user U_i
$\lvert g_i[k]\rvert$	The absolute value of $g_i[k]$
$[g_i]$	The gradient ciphtext of user U_i
$[\lvert g_i\rvert]$	Absolute gradient ciphtext of user U_i
$[\lvert g_{i,j}^{+}\rvert]$	The homomorphic sum of absolute gradient ciphertexts $[\lvert g_i\rvert], [\lvert g_j\rvert]$
$[\widetilde{g_{i,j}}]$	The absolute gradient ciphertext homomorphic sum on the elements of $[\lvert g_{i,j}^{+}\rvert]$
$[\lvert g_{i,j}^{-}\rvert]$	The homomorphic sum of difference between absolute gradient ciphertexts $[\lvert g_i\rvert], [\lvert g_j\rvert]$
$[r_c]$	Encrypted mask
$[r_c\lvert g_{i,j}^{-}\rvert]$	The blinded difference ciphertext
$r_c\lvert g_{i,j}^{-}\rvert$	The blinded difference
$sign_{i,j}$	The sign vector of all elements in $r_c\lvert g_{i,j}^{-}\rvert$
$[\lvert g_{i,j}^{*}\rvert]$	The corrected absolute gradient difference ciphertext
$[\lvert\widehat{g_{i,j}}\rvert]$	The homomorphic sum of the elements in absolute gradient difference ciphertext $[\lvert g_{i,j}^{*}\rvert]$
$[G]$	The aggregated gradient ciphertext
C_i	Confidence score of user U_i
θ	Dynamic threshold
$reputation_i$	Reputation score of user U_i

B. Federated learning

Assume that there are N users $U_i(i \in \{1, 2, \ldots, N\})$ participating in federated learning, each possessing private dataset D_i.

In the initialization phase, the server initializes the global model ω^0 and distributes it to all users. During the t-th training round, user U_i updates the global model ω^{t-1} using their private dataset D_i and calculates the local gradient $g_i = \nabla_\omega L(D_i, \omega^{t-1})$, where L denotes the loss function, $\nabla_\omega L$ denotes the partial derivative of the loss function L with respect to the model ω and ω^{t-1} represents the global model. The local gradient g_i is then uploaded to the server.

In the aggregation phase, the server calculates the global gradient via averaging: $G = \frac{1}{N}\sum_{i=1}^{N} g_i$. The aggregated global gradient G is subsequently sent back to all users.

In the model update phase, each user $U_i(i \in \{1, 2, \ldots, N\})$ updates their global model ω^t using the global gradient G: $\omega^t = \omega^{t-1} - \mu\frac{G}{N}$, where μ is the learning rate.

This cycle of local training, gradient aggregation, and model updating is repeated until a predefined stopping condition is met.

C. Cheon-Kim-Kim-Song (CKKS)

CKKS [35] is a lattice-based fully homomorphic encryption scheme that supports both addition and multiplication over encrypted data. It enables secure approximate computation on floating-point numbers and multi-dimensional vectors by first encoding them into plaintext polynomials, performing homomorphic operations on the encrypted polynomials, and then decoding and decrypting to recover approximate results.

The plaintext and ciphertext spaces of CKKS are defined over the polynomial ring $R_q = \mathbb{Z}_q[X]/(X^M + 1)$, where $M = 2^K$ is the ring dimension, q is the modulus, $\mathbb{Z}_q[X]$ represents the polynomial ring whose coefficients are integer modulo q, and $X^M + 1$ is the polynomial modulo which all arithmetic operations in the ring are performed. The key distribution χ and error distribution Ψ are defined over the ring R_q, with samples $t, v \leftarrow \chi, a \in R_q$ chosen uniformly at random,and $e, e_0, e_1, e' \leftarrow \Psi$. Approximate operations are performed on multi-dimensional vectors whose elements may be real or complex numbers. A vector $\mathbf{z} \in C^{M/2}$ is first encoded into a plaintext polynomial using canonical embedding and the fast Fourier transform, and scaled by a factor s. Homomorphic operations are then performed on the encrypted polynomial, and the decoding process applies the inverse transformation to recover an approximate version of the original vector, with the error controlled by the scaling factor s. The key algorithms in CKKS are as follows:

$\cdot\boldsymbol{KenGen(1^\lambda) \rightarrow (pk, sk, evk)}$: Given the security parameter λ, let $P = P(\lambda, q)$, and sample a' uniformly at random from R_{Pq}. This algorithm generates a public key $pk = (b, a) \in R_q^2$, where $b = -a * s + e$, a secret key $sk = (1, t)$, and an evaluation key $evk = (a', b')$, where $b' = -a' * s + e' + Ps^2$.

$\cdot\boldsymbol{Ecd(z, s) \rightarrow (m(x))}$: Given an $(M/2)$-dimensional vector $\mathbf{z}$ and a scaling factor s, this algorithm encodes $\mathbf{z}$ into a plaintext polynomial $m(x) \in R_q$ by applying the fast Fourier transform and scaling the coefficients by s.

$\cdot\boldsymbol{Enc(m(x), pk) \rightarrow c}$: Given a plaintext polynomial $m(x)$ and a public key pk, this algorithm encrypts $m(x)$ to produce a ciphertext $c = (c_0, c_1) = (v*b+m(x)+e_0, v*a+e_1) \in R_q^2$.

$\cdot\boldsymbol{Dec(c, sk) \rightarrow m(x)}$: Given a ciphertext $c = (c_0, c_1)$ and a secret key sk, this algorithm decrypts c to recovers the plaintext polynomial $m(x) = c * sk = c_0 + c_1 * t$.

$\cdot\boldsymbol{Dcd(m(x), s) \rightarrow z}$: Given a plaintext polynomial $m(x)$ and scaling factor s, this algorithm decodes $m(x)$ into a vector $\mathbf{z}$ utilizing the inverse fast Fourier transform.

CKKS supports computations on ciphertexts, including addition and multiplication. Specifically:

$\cdot\boldsymbol{Add(c, c^*) \rightarrow c_{Add}}$: Given ciphertexts $c, c^* \in R_q^2$, this algorithm returns their sum $c_{Add} = c \oplus c^*$.

$\cdot\boldsymbol{Mult(c, c^*, evk) \rightarrow c_{Mult}}$: Given ciphertexts $c, c^* \in R_q^2$ and an evaluation key evk, this algorithm returns the product $c_{Mult} = c \otimes c^*$ in the encrypted form.

D. Bray-Curtis dissimilarity

The Bray-Curtis dissimilarity [36], proposed by Bray and Curtis, is a metric used to measure differences in species abundance between different regions. It has been extended to various domains, including ecological studies and data analysis, for quantifying dissimilarities in multi-dimensional data. The computation of Bray-Curtis dissimilarity is as follow:

$$D_{BC} = \frac{\sum_{i=1}^{p} |x_i - y_i|}{\sum_{i=1}^{p} (x_i + y_i)},$$

where $x_i \geq 0$ and $y_i \geq 0$ are the i-th elements of vectors x and y, respectively, and p is the dimension of the vectors. Given that both x_i and y_i are non-negative, it follows that $D_{BC} \in [0, 1]$. Specifically, when the two vectors are identical, $D_{BC} = 0$ (indicating complete similarity), and when the vectors are maximally dissimilar, $D_{BC} = 1$ (indicating complete dissimilarity). In general, a value closer to 0 implies greater similarity between the vectors, whereas a value closer to 1 indicates a greater difference.

IV. PROBLEM STATEMENT

A. System model

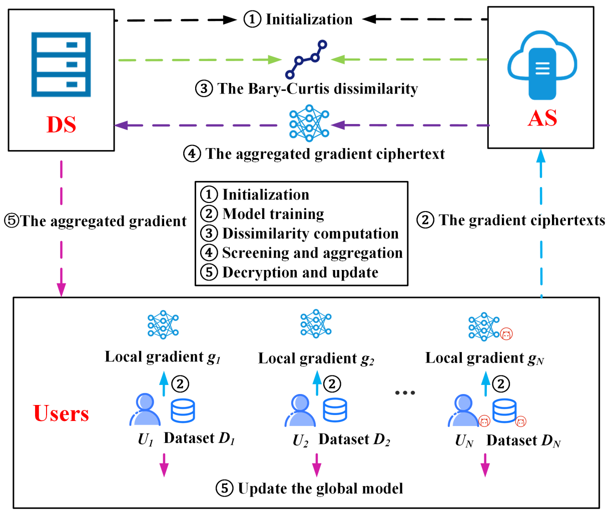


Fig. 1: system model

As shown in Fig. 1, BPFLH consists of three entities:

·Users: Users train the model with their private datasets to compute local gradients, encrypt the local gradients, upload the gradient ciphertexts to AS, and update their local model based on the aggregated gradient sent by the DS.

·Aggregation Server (AS): AS cooperates with DS to calculate the Bray-Curtis dissimilarity between the users' local gradients, screens Byzantine users, and aggregates the gradient ciphertexts of benign users.

·Decryption Server (DS): DS works with AS to compute the Bray-Curtis dissimilarity between the users' local gradients, and decrypts the aggregated gradient ciphertext.

The system workflow is as follows:

Step ①: AS and DS initialize the system parameters.

Step ②: Each user performs local training using their private datasets to calculate the local gradients. These local gradients are encrypted and the gradient ciphertexts are uploaded to AS.

Step ③: AS, in collaboration with DS, calculates the Bray-Curtis dissimilarity between the users' local gradients.

Step ④: AS selects users based on the Bray-Curtis dissimilarity, applies penalties to Byzantine users, and aggregates the gradient ciphertexts of benign users. The aggregated gradient ciphertext is forwarded to DS.

Step ⑤: DS decrypts the aggregated gradient ciphertext to obtain the aggregated gradient, which is then returned to all users. Users employ the aggregated gradient to update their global model.

B. Threat model

In BPFLH, we adopt the following threat model:

·Byzantine Attacks in users: Similar to prior works [12], [34], [37], we consider a robust threat model in which no more than $N/2$ users in the federated learning system are either compromised or under the control of a malicious adversary, where N represents the total number of users. These malicious users may launch Byzantine attacks, such as label flipping or model poisoning. Specifically, the adversary may alter the labels of their private data, introducing corrupted data into the training process, resulting in incorrect local gradients. Alternatively, the adversary may not modify the data itself but instead directly manipulate the local gradients after the users have trained on clean data. The goal of the adversary is to manipulate the global model's training by introducing incorrect gradients or misleading data, ultimately degrading the model's performance.

·Honest-but-Curious and Non-Collusive Servers: Following the previous works [12], [16], [37], we assume that both AS and DS are non-colluding and honest-but-curious. While they will perform the federated learning protocol as prescribed, they may be interested in users' private data and could attempt to extract private information from the data they have access to. However, they will not intentionally manipulate or disrupt the federated learning process.

Note: In machine learning, the dual-server model with the assumption of non-collusion is typically considered as a weaker assumption compared to the single-server model. However, this architecture offers two significant advantages for federated learning systems: 1) it greatly reduces the communication overhead between users and servers. 2) it enhances the robustness and flexibility of the system. As a result, the non-colluding dual-server setup has been extensively adopted in previous works, such as LSFL [12], ShieldFL [16], PPARFL [37] and PPFUCCR [38].

C. Design goals

In response to the aforementioned threat model, BPFLH should meet the following design goals:

·Fidelity: BPFLH ensures that, in the absence of attacks, the global model's accuracy remains comparable to FedAvg, a standard federated learning algorithm. It also guarantees that the privacy-preserving and Byzantine user detection mechanisms do not degrade model's accuracy.

·Robustness: BPFLH effectively distinguishes between malicious gradients and benign gradients caused by data heterogeneity, preventing the misclassification of benign gradients as malicious. This ensures the model's robustness in non-IID settings and defends against Byzantine attacks, accurately identifying and excluding malicious gradients.

·Privacy Preservation: BPFLH guarantees that users' privacy remains protected and is not disclosed to AS, DS, or Byzantine users.

V. Proposed Scheme

A. Overview

In federated learning, the heterogeneous data environment causes local gradients from users with different data distributions to differ significantly. These variations are natural and do not indicate malicious behavior, whereas Byzantine users may manipulate their gradients in different ways to disrupt the global model. Such manipulations can be subtle, aiming at targeted outputs, or large and abrupt, aiming to generally disrupt training.

To accurately distinguish malicious gradients from benign deviations caused by data heterogeneity, BPFLH employs Bray-Curtis dissimilarity to compute element-wise dissimilarity between local gradients. Unlike existing methods that rely on global dissimilarity-based metrics, Bray–Curtis dissimilarity detects differences at the element level, allowing it to capture subtle gradient variations more effectively and reducing the risk of misclassifying benign deviations as Byzantine attacks.

Since Bray-Curtis dissimilarity originates from ecological diversity metrics, where the number of species is non-negative, we take the absolute value of the gradient element before calculation to ensure that all values are non-negative. The Bray-Curtis dissimilarity between the local gradients of users is defined as:

$$BC_{i,j} = \frac{\sum_{k=1}^{d} ||g_i[k]| - |g_j[k]||}{\sum_{k=1}^{d} (|g_i[k]| + |g_j[k]|)}, \tag{1}$$

where d is the dimensionality of the gradients g_i, g_j. It is worth noting that the Bray-Curtis dissimilarity is symmetric, i.e., $BC_{i,j} = BC_{j,i}$. Therefore, it is sufficient to compute $BC_{i,j}$ only for pairs where $i > j$, which reduces computational cost without affecting the results. This element-wise comparison makes it particularly effective at quantifying dissimilarity, especially when the element values are small or significantly different, providing a robust basis for distinguishing malicious and benign gradients, even under non-IID data settings.

To protect user privacy, CKKS homomorphic encryption is applied during the computation, ensuring that sensitive information is not exposed. This design enables the AS to perform Byzantine user detection directly over encrypted gradients, without compromising users' privacy. Specifically, the Bray–Curtis dissimilarity is computed over the absolute gradient ciphertext elements $[|g_i[k]|]$ and $[|g_j[k]|]$. Specifically, we need to compute: $\sum_{k=1}^{d} ([|g_i[k]|] + [|g_j[k]|])$ and $\sum_{k=1}^{d} |[|g_i[k]|] - [|g_j[k]|]|$. The homomorphic sum $\sum_{k=1}^{d} ([|g_i[k]|] + [|g_j[k]|])$ can be directly calculated using homomorphic addition and homomorphic summation operations supported by CKKS. For the detailed calculations, refer to dissimilarity computation phase (3)-a) of section V-B. However, computing $\sum_{k=1}^{d} |[|g_i[k]|] - [|g_j[k]|]|$ presents a key challenge, as CKKS encryption cannot directly perform subtraction and absolute value operations on encrypted data. To address this challenge, we design a novel privacy-preserving computation method. Specifically, AS performs homomorphic multiplication on $[|g_j[k]|]$ and the encrypted -1, obtaining $[-|g_j[k]|]$. Then, AS performs homomorphic addition on $[|g_i[k]|]$ and $[-|g_j[k]|]$ to compute $[|g_i[k]|] - [|g_j[k]|]$. To determine the sign of $[|g_i[k]|] - [|g_j[k]|]$ while preserving privacy, AS blinds each difference $[|g_i[k]|] - [|g_j[k]|]$ with a unique encrypted positive mask $[r_c]$ and performs homomorphic multiplication. The resulting blinded values are then securely decrypted by DS to extract the sign vector $sign$. Since r_c is a positive, the blinding will not change the sign. The corrected absolute value $|[|g_i[k]|] - [|g_j[k]|]|$ is computed by performing homomorphic multiplication on the encrypted sign vector $[sign]$ and $[|g_i[k]|] - [|g_j[k]|]$. $\sum_{k=1}^{d} |[|g_i[k]|] - [|g_j[k]|]|$ can be obtained through homomorphic summation. Finally, after decrypting $\sum_{k=1}^{d} ([|g_i[k]|] + [|g_j[k]|])$ and $\sum_{k=1}^{d} |[|g_i[k]|] - [|g_j[k]|]|$, DS can calculate Bray-Curtis dissimilarity between local gradients of users. For the detailed calculations, refer to dissimilarity computation phase-steps (3)-b) to g) of section V-B.

Based on the computed Bray-Curtis dissimilarities, AS derives a confidence score C_i for each user. A dynamic threshold is determined using the median $\overline{C}$ and standard deviation σ of all users' confidence scores to distinguish benign users from Byzantine ones. Unlike threshold-based approaches, which may discard benign gradients in non-IID data settings, BPFLH integrates a reputation-based penalty mechanism. Byzantine users are penalized gradually, while benign users are protected from misjudgment. AS aggregates the gradient ciphertexts of all benign users and delivers the aggregated gradient ciphertext to DS. DS decrypts the aggregated gradient ciphertext and then transmits the aggregated gradient to all users.

B. Construction of BPFLH

BPFLH contains the following five phases: initialization, model training, dissimilarity computation, screening and aggregation, and decryption and update.

1) Initialization

Assume that there are N users participating in federated learning, denoted by the set $U = \{U_i\}_{i\in\{1,2,\ldots,N\}}$.

a) Key generation and distribution: Given a security parameter λ, DS produces the public key pk, private key sk, and evaluation key evk. The public key pk is broadcasted to each user $\{U_i\}_{i\in\{1,2,\ldots,N\}}$, while the private key sk and evaluation key evk are stored locally.

b) Federated learning initialization: AS initializes the global model ω^0, defines the loss function L, sets the learning rate μ and the scaling factor s, and randomly generates C_N^2 positive real-valued masks $r_c (c \in [1, C_N^2], r_c > 0, r_c \in \mathbb{R})$. Subsequently, the global model ω^0, loss function L, learning rate μ, and the scaling factor s are broadcasted to all users $U_i \in U$, while the scaling factor s is also delivered to DS. Additionally, AS initializes the reputation score $reputation_i$ for each user $\{U_i\}_{i\in\{1,2,\ldots,N\}}$ and specifies the punishment factor α.

2) Model Training

a) Gradient calculation: In the t-th round of training, user U_i employs its private dataset D_i to train the global model ω^{t-1}, obtaining the corresponding local gradient g_i of dimension d:

$$g_i = \nabla_\omega L(D_i, \omega^{t-1}), \tag{2}$$

where L represents the loss function.

b) Absolute gradient calculation: User U_i calculates the absolute value of each element $g_i[k]$ in their local gradient g_i, producing the element-wise absolute gradient, hereinafter referred to as the absolute gradient and denoted $|g_i|$, i.e., a vector whose k-th element is $|g_i[k]|$.

c) Gradient and absolute gradient encryption: User U_i encrypts both the local gradient g_i and its absolute gradient $|g_i|$ using the public key pk, obtaining the gradient ciphertext $[g_i]$ and absolute gradient ciphertext $[|g_i|]$:

$$[g_i] = Enc(Ecd(g_i, s), pk) \tag{3}$$

$$[|g_i|] = Enc(Ecd(|g_i|, s), pk). \tag{4}$$

d) Ciphertext upload: User U_i delivers the gradient ciphertext $[g_i]$ and absolute gradient ciphertext $[|g_i|]$ to AS.

The detailed procedures for model training are summarized in Algorithm 1.

Algorithm 1 Model Training

```
1: Input: Private dataset $D_i$ of user $U_i$, global model $\omega^{t-1}$
   and public key $pk$.
2: Output: Gradient ciphertext $[g_i]$ and absolute gradient
   ciphertext $[|g_i|]$.
3: for each user $U_i$ in $U$ do
4:     $U_i$ computes local gradient $g_i = \nabla_\omega L(D_i, \omega^{t-1})$ and
       absolute gradient $|g_i|$.
5:     $U_i$ encrypts $g_i$ and $|g_i|$ with $pk$ to obtain ciphertexts
       $[g_i]$ and $[|g_i|]$, then sends them to AS.
6: end for
```

3) Dissimilarity Computation

a) Computation of homomorphic sum of absolute gradient ciphertext elements: For each user pair $(U_i, U_j)(i, j \in \{1, 2, \ldots, N\}, i > j)$, AS calculates the homomorphic sum $[|g_{i,j}^+|]$ of their uploaded absolute gradient ciphertexts $[|g_i|], [|g_j|]$ as follows:

$$[|g_{i,j}^+|] = Add([|g_i|], [|g_j|]); \tag{5}$$

Then AS performs homomorphic summation on the elements $[|g_{i,j}^+[k]|]$ of $[|g_{i,j}^+|]$ to obtain the absolute gradient ciphertext homomorphic sum $[\widetilde{g_{i,j}}]$:

$$[\widetilde{g_{i,j}}] = \sum_{k=1}^{d} [|g_{i,j}^+[k]|], \tag{6}$$

Here, $\sum$ denotes homomorphic summation, implemented via repeated CKKS addition operations, since ordinary arithmetic cannot be applied to encrypted elements.

b) Calculation of homomorphic sum of absolute gradient difference ciphertext: Similarly, for each user pair $(U_i, U_j)(i, j \in \{1, 2, \ldots, N\}, i > j)$, AS calculates the homomorphic sum $[|g_{i,j}^-|]$ of difference between their absolute gradient ciphertexts $[|g_i|], [|g_j|]$:

$$[|g_{i,j}^-|] = Add([|g_i|], Mult(Enc(Ecd(-1, s), pk), [|g_j|])) \tag{7}$$

Here, $Mult(Enc(Ecd(-1,s),pk),[|g_j|])$ homomorphically multiplies $[|g_j|]$ with the encrypted -1, producing the negative absolute gradient ciphertext $[-|g_j|]$.$[|g^-_{i,j}|]$ is hereafter referred to as the absolute gradient difference ciphertext.

c) Blinding: To blind each absolute gradient difference ciphertext $[|g^-_{i,j}|]$, AS computes a unique encrypted mask $[r_c] = Enc(Ecd(r_c,s),pk)$ for each user pair $(U_i,U_j)(i,j \in \{1,2,\ldots,N\}, i > j)$, then calculates the blinded difference ciphertext $[r_c|g^-_{i,j}|]$:

$$[r_c|g^-_{i,j}|] = Mult([r_c],[|g^-_{i,j}|],evk), \tag{8}$$

which is sent to DS.

d) Decryption: DS decrypts and decodes the blinded difference ciphertext $[r_c|g^-_{i,j}|]$ to obtain the blinded difference $r_c|g^-_{i,j}|$:

$$r_c|g^-_{i,j}| = Dcd(Dec([r_c|g^-_{i,j}|],sk),s). \tag{9}$$

DS extracts the sign of each element in $r_c|g^-_{i,j}|$, stores them in a sign vector $sign_{i,j}$, and delivers this sign vector $sign_{i,j}$ to AS.

e) Sign-based correction: AS encrypts the sign vector $sign_{i,j}$, resulting in the encrypted sign vector $[sign_{i,j}]$:

$$[sign_{i,j}] = Enc(Ecd(sign_{i,j},s),pk). \tag{10}$$

Then AS homomorphically multiplies the sign vector $[sign_{i,j}]$ with the absolute gradient difference ciphertext $[|g^-_{i,j}|]$ to produce the corrected absolute gradient difference ciphertext $[|g^*_{i,j}|]$:

$$[|g^*_{i,j}|] = Mult([sign_{i,j}],[|g^-_{i,j}|],evk). \tag{11}$$

This operation flips the sign of each element in $[|g^-_{i,j}|]$ whose decrypted value is negative, ensuring that all elements in the corrected absolute gradient difference ciphertext $[|g^*_{i,j}|]$ are non-negative.

f) Computation of homomorphic sum of corrected absolute gradient difference ciphertext elements: AS performs homomorphic summation on the elements $[|g^*_{i,j}|[k]]$ of $[|g^*_{i,j}|]$, obtaining the homomorphic sum $[|\widehat{g_{i,j}}|]$ of absolute gradient difference ciphertext elements $[|g^*_{i,j}|[k]]$:

$$[|\widehat{g_{i,j}}|] = \sum_{k=1}^{d}[|g^*_{i,j}|[k]]. \tag{12}$$

AS delivers both gradient ciphertext homomorphic sum $[\widetilde{g_{i,j}}]$ and the homomorphic sum $[|\widehat{g_{i,j}}|]$ of the absolute gradient difference ciphertext elements to DS.

g) Calculation of Bray-Curtis dissimilarity: DS decrypts and decodes $[\widetilde{g_{i,j}}]$ and $[|\widehat{g_{i,j}}|]$, obtaining the numerator and denominator required for calculating the Bray-Curtis dissimilarity:

$$|\widehat{g_{i,j}}| = Dcd(Dec(([|\widehat{g_{i,j}}|],sk),s) \tag{13}$$

$$\widetilde{g_{i,j}} = Dcd(Dec([\widetilde{g_{i,j}}],sk),s) \tag{14}$$

The Bray-Curtis dissimilarity between the local gradients of users U_i and U_j is then calculated as:

$$BC_{i,j} = \frac{|\widehat{g_{i,j}}|}{\widetilde{g_{i,j}}}. \tag{15}$$

DS delivers $\{BC_{i,j}\}_{i,j\in[1,N],i>j}$ to AS.

The procedures for calculating the Bray-Curtis dissimilarity are detailed in Algorithm 2.

Algorithm 2 Dissimilarity Computation

```
1:  Input: Public key $pk$, evaluation key $evk$, gradient ciphertext $[g_i]$, scaling factor $s$ and absolute gradient ciphertext $[|g_i|]$.
2:  Output: Bray-Curtis dissimilarity $BC_{i,j}$.
3:  for each user pair $(U_i,U_j)$, $(i > j)$ do
4:      AS computes the homomorphic sum $[|g^+_{i,j}|] = Add([|g_i|],[|g_j|])$ of absolute gradient ciphertexts $[|g_i|],[|g_j|]$ and computes the absolute gradient ciphertext homomorphic sum $[\widetilde{g_{i,j}}] = \sum_{k=1}^{d}[|g^+_{i,j}[k]|]$.
5:      AS computes the homomorphic sum of difference between $[|g_i|]$ and $[|g_j|]$: $[|g^-_{i,j}|] = Add([|g_i|], Mult(Enc(Ecd(-1,s),pk),[|g_j|]))$.
6:      AS computes the encrypted mask $[r_c] = Enc(Ecd(r_c,s),pk)$ with $pk$ and the blinded difference ciphertext $[r_c|g^-_{i,j}|] = Mult([r_c],[|g^-_{i,j}|],evk)$ with $evk$, and sends $[r_c|g^-_{i,j}|]$ to DS.
7:      DS decrypts and decodes $[r_c|g^-_{i,j}|]$ to obtain $r_c|g^-_{i,j}| = Dcd(Dec([r_c|g^-_{i,j}|],sk),s)$, extracts the sign vector $sign_{i,j}$ of $r_c|g^-_{i,j}|$, and sends it to AS.
8:      AS encrypts $sign_{i,j}$ with $pk$ to get the encrypted sign vector $[sign_{i,j}] = Enc(Ecd(sign_{i,j},s),pk)$.
9:      AS performs homomorphic multiplication on $[sign_{i,j}]$ and $[|g^-_{i,j}|]$ to obtain the corrected absolute gradient difference ciphertext $[|g^*_{i,j}|] = Mult([sign_{i,j}],[|g^-_{i,j}|],evk)$.
10:     AS calculates the homomorphic sum of absolute gradient difference ciphertext elements $[|g^*_{i,j}|[k]]$: $[|\widehat{g_{i,j}}|] = \sum_{k=1}^{d}[|g^*_{i,j}|[k]]$.
11:     AS delivers $[\widetilde{g_{i,j}}]$ and $[|\widehat{g_{i,j}}|]$ to DS.
12:     DS decrypts $[\widetilde{g_{i,j}}]$ and $[|\widehat{g_{i,j}}|]$ to obtain $|\widehat{g_{i,j}}| = Dcd(Dec(([|\widehat{g_{i,j}}|],sk),s)$ and $\widetilde{g_{i,j}} = Dcd(Dec([\widetilde{g_{i,j}}],sk),s)$
13:     DS computes Bray-Curtis dissimilarity $BC_{i,j} = \frac{|\widehat{g_{i,j}}|}{\widetilde{g_{i,j}}}$ and delivers it to AS.
14: end for
```

4) Screening and Aggregation

a) Computation of dynamic threshold and user filtering: Based on the set $\{BC_{i,j}\}_{i,j\in[1,N],i>j}$ received from the DS, AS can derive the Bray-Curtis dissimilarities between the local gradients of each user U_i and all other users U_j $(j \in [1,N], j \neq i)$ based on the fact that $BC_{i,j} = BC_{j,i}$. AS calculates the average Bray-Curtis dissimilarity between the local gradients of user U_i and all other users U_j $(j \in [1,N], j \neq i)$, which is defined as the confidence score of user U_i:

$$C_i = \frac{\sum_{j=1,j\neq i}^{N} BC_{i,j}}{N-1}. \tag{16}$$

Next, AS further calculates the median $\overline{C}$ and standard deviation σ of all users' confidence scores, and sets the dynamic threshold $\theta = \overline{C} + m * \sigma$, where $m \in (0,1)$ is a floating-point parameter, typically assigned a small value, that provides error tolerance and can be dynamically adjusted as needed. Users whose confidence scores exceeding the dynamic threshold θ are identified as Byzantine users U_i' (the set denoted U'), while the remaining users are classified as benign users U_i'' (the set denoted U'').

b) Punishment: For each Byzantine user $U_i'(U_i' \in U')$, AS verifies their reputation score $reputation_i$. If $reputation_i < 0$, the user is removed from federated learning process; otherwise, the reputation is penalized by subtracting the punishment factor α:

$$reputation_i = reputation_i - \alpha \tag{17}$$

c) Gradient aggregation: AS aggregates the gradient ciphertexts $[g_i]$ of all benign users $U_i''(U_i'' \in U'')$, producing the aggregated gradient ciphertext $[G]$:

$$[G] = \sum_{U_i'' \in U''} [g_i] \tag{18}$$

where $|U''|$ denotes the number of benign users. AS delivers the aggregated gradient ciphertext $[G]$ to DS, and broadcasts the number of benign users $|U''|$ to all users $U_i \in U$.

Algorithm 3 presents the screening and aggregation procedures.

Algorithm 3 Screening and Aggregation

```
1:  Input: Bray-Curtis dissimilarities $\{BC_{i,j}\}_{i,j \in [1,N], i>j}$.
2:  Output: Aggregated gradient ciphertext $[G]$ and the number of benign users $|U''|$.
3:  for each user $U_i$ in $U$ do
4:      AS computes the confidence score $C_i = \frac{\sum_{j=1,j\neq i}^{N} BC_{i,j}}{N-1}$.
5:  end for
6:  AS computes the median $\overline{C}$ and standard deviation $\sigma$ of all users' confidence scores.
7:  AS calculates dynamic threshold $\theta = \overline{C} + m \cdot \sigma$.
8:  AS identifies Byzantine users $U_i'$ with $C_i > \theta$ and benign users $U_i''$ with $C_i \leq \theta$.
9:  for each Byzantine user $U_i'$ in $U'$ do
10:     AS verifies the reputation score $reputation_i$.
11:     if $reputation_i < 0$ then
12:         Remove $U_i'$.
13:     else
14:         $reputation_i = reputation_i - \alpha$
15:     end if
16: end for
17: for each benign user $U_i''$ in $U''$ do
18:     AS aggregates the gradient ciphertexts as: $[G] = \sum_{U_i'' \in U''} [g_i]$.
19: end for
20: AS delivers the aggregated gradient ciphertext $[G]$ to DS and broadcasts the number of benign users $|U''|$ to all users $U_i \in U$.
```

5) Decryption and Update

a) Decryption and distribution: DS decrypts and decodes the aggregated gradient ciphertext $[G]$ to obtain the aggregated gradient G:

$$G = Dcd(Dec([G], sk), s), \tag{19}$$

DS transmits the aggregated gradient G to all users $U_i \in U$.

b) Model update: Each user U_i updates its global model using the aggregated gradient as follows:

$$\omega^t = \omega^{t-1} - \mu \frac{G}{|U''|}. \tag{20}$$

Training proceeds to the next round until the model converges or a stopping condition is met.

The detailed procedures for decryption and update are summarized in Algorithm 4.

Algorithm 4 Decryption and Model Update

```
1: Input: Aggregated gradient ciphertext $[G]$, private key $sk$, scaling factor $s$, global model $\omega^{t-1}$, learning rate $\mu$, and the number of benign users $|U''|$.
2: Output: Global model $\omega^t$.
3: DS decrypts and decodes the aggregated gradient ciphertext $[G]$ as $G = Dcd(Dec([G], sk), s)$ and transmits the aggregated gradient $G$ to all users $U_i \in U$.
4: for each user $U_i$ in $U$ do
5:     Updates global model $\omega^t = \omega^{t-1} - \mu \frac{G}{|U''|}$
6: end for
```

VI. Theoretical Analysis

Theorem 1: AS, DS, and Byzantine users cannot obtain the private information of benign users.

Proof: In BPFLH, each user $U_i(i \in [1, N])$ encrypts its local gradient g_i and the absolute gradient $|g_i|$ using the CKKS homomorphic encryption scheme, and delivers the gradient ciphertext $[g_i]$ and absolute gradient ciphertext $[|g_i|]$ to AS. AS calculates the blinded difference ciphertext $[r_c|g_{i,j}^-|]$ $(i, j \in [1, N], i > j)$, the gradient ciphertext homomorphic sum $[\widetilde{g_{i,j}}]$, the homomorphic sum $[|\widehat{g_{i,j}}|]$ of the absolute gradient difference ciphertext elements and the aggregated gradient ciphertext $[G]$, and transmits them to DS. DS computes the sign vector $sign_{i,j}$ $(i, j \in [1, N], i > j)$ and Bray-Curtis dissimilarity $BC_{i,j}$, and returns them to AS. Furthermore, DS also decrypts the aggregated gradient ciphertext $[G]$, and delivers the aggregated gradient G to all users. From the above processes, we can observe that AS knows the gradient ciphertext $[g_i]$, the absolute gradient ciphertext $[|g_i|]$, the sign vectors $sign_{i,j}$ and the Bray-Curtis dissimilarity $BC_{i,j}$. DS knows the blinded difference ciphertext $[r_c|g_{i,j}^-|]$, the homomorphic sum $[|\widehat{g_{i,j}}|]$ of the absolute gradient difference ciphertext elements, and the gradient ciphertext homomorphic sum $[\widetilde{g_{i,j}}]$. Byzantine users know the aggregated gradient G. Since AS and DS are assumed to be non-colluding, they cannot combine their information. Therefore, we analyze their ability to infer users' private information separately.

For AS, it cannot obtain the users' private information from the data it possesses. The ciphertexts $[g_i]$ and $[|g_i|]$ are both encrypted under CKKS homomorphic encryption scheme, which ensures semantic security against chosen-plaintext attacks and provides functional privacy [39]. The private key sk is exclusively held by DS. Thus, without the private key sk, AS cannot decrypt these ciphertexts. Moreover, Bray-Curtis dissimilarity $BC_{i,j}$ contains only scalar values, while each local gradient g_i is a vector. Hence, AS cannot infer users' local gradients from these scalar metrics. The sign vectors $sign_{i,j}$ merely record element-wise ordering relations between gradients of user pairs and do not contain actual gradient values. Consequently, AS cannot infer users' private information from the sign vectors $sign_{i,j}$. For DS, it cannot obtain users' private information from the homomorphic sum $[|\widehat{g_{i,j}}|]$ of the absolute gradient difference ciphertext elements and the gradient ciphertext homomorphic sum $[\widetilde{g_{i,j}}]$. After decryption using the private key sk, DS only obtains the numerator and denominator required for calculating the Bray-Curtis dissimilarity, both of which are two scalar values that reveal no individual gradient components. In addition, the blinded difference ciphertext $[r_c|g_{i,j}^-|]$ is blinded by a random mask r_c, which is known only to AS. Even if DS can decrypt $[r_c|g_{i,j}^-|]$, it obtains only $r_c|g_{i,j}^-|$. Without knowledge of r_c, DS cannot recover $|g_{i,j}^-|$ or infer any gradient information. For Byzantine users, it cannot extract or infer any benign user's local gradient from the aggregated gradient G as G is the aggregation of all benign users' gradients and does not expose any individual gradient.

Therefore, BPFLH ensures that the private information of benign users is protected from non-colluding AS and DS and the Byzantine users.

Theorem 2: Under the assumption that AS and DS do not collude, our dual-server model can strictly preserves users' private.

Proof: During the execution of the protocol by AS, the real view of AS is defined as $REAL_{AS} = (pk, [g_i], [|g_i|], r_c, [r_c], [|g_{i,j}^+|], [|g_{i,j}^-|], [r_c|g_{i,j}^-|], sign_{i,j}, [|g_{i,j}^*|], [\widetilde{g_{i,j}}], [|\widehat{g_{i,j}}|], BC, reputation_i, [G])$.

Let S_{AS} be a probabilistic polynomial-time simulator that constructs the ideal-world view of AS. The ideal view simulated by S_{AS} is $IDEAL_{AS} = (pk, [g_i^*], [|g_i^*|], r_c, [r_c^*], [|g_{i,j}^{+*}|], [|g_{i,j}^{-*}|], [r_c|g_{i,j}^{-*}|], sign_{i,j}, [|g_{i,j}^{**}|], [\widetilde{g_{i,j}}^*], [|\widehat{g_{i,j}}|^*], BC, reputation_i, [G^*])$. where all elements marked with "*" are random ciphertexts generated by the simulator S_{AS}. To prove that $REAL_{AS}$ and $IDEAL_{AS}$ are computationally indistinguishable, we construct the following hybrid sequence:

$\mathbf{Hyb_0}$: Real protocol execution, $Hyb_0 = REAL_{AS}$.

$\mathbf{Hyb_1}$: Replace the user-uploaded ciphertexts $[g_i]$ and $[|g_i|]$ with random encryptions $[g_i^*] \leftarrow Enc(0, pk)$ and $[|g_i^*|] \leftarrow Enc(0, pk)$. Since CKKS homomorphic encryption scheme is semantically security, this replacement is computationally indistinguishable.

$\mathbf{Hyb_2}$: Replace $[g_{i,j}^+], [|g_{i,j}^-|], [r_c|g_{i,j}^-|], [|g_{i,j}^*|], [\widetilde{g_{i,j}}], [|\widehat{g_{i,j}}|]$ with random encryptions $[|g_{i,j}^{+*}|], [|g_{i,j}^{-*}|], [r_c|g_{i,j}^{-*}|], [|g_{i,j}^{**}|], [\widetilde{g_{i,j}}^*], [|\widehat{g_{i,j}}|^*]$. By the semantic security of CKKS homomorphic encryption scheme, this replacement is computationally indistinguishable.

$\mathbf{Hyb_3}$: Replace the final output ciphertext $[G]$ with random encryption $[G^*]$. Similarly, due to the semantic security of CKKS homomorphic encryption scheme, this replacement is computationally indistinguishable.

At this point $Hyb_3 = IDEAL_{AS}$, thus the view of AS generated by the simulator S_{AS} in the ideal world is computationally indistinguishable from the view of AS in the real world.

Similarly, the real view of DS is defined as:$REAL_{DS} = (sk, evk, [r_c|g_{i,j}^-|], [\widetilde{g_{i,j}}], [|\widehat{g_{i,j}}|], [G], r_c|g_{i,j}^-|, sign_{i,j}, \widetilde{g_{i,j}}, |\widehat{g_{i,j}}|, BC, G)$.

Let S_{DS} be a probabilistic polynomial-time simulator that constructs the ideal-world view of DS. The ideal view simulated by S_{DS} is $IDEAL_{DS} = (sk, evk, [r_c|g_{i,j}^{-*}|], [\widetilde{g_{i,j}}^*], [|\widehat{g_{i,j}}|^*], [G^*], r_c|g_{i,j}^{-*}|, sign_{i,j}, \widetilde{g_{i,j}}^*, |\widehat{g_{i,j}}|^*, BC, G)$. where all elements marked with "*" are random ciphertexts generated by the simulator S_{DS}. The indistinguishability is shown through the following hybrid sequence:

$\mathbf{Hyb_0}$: Real protocol execution, $Hyb_0 = REAL_{DS}$.

$\mathbf{Hyb_1}$: Replace $r_c|g_{i,j}^-|$ with a random value $r_c|g_{i,j}^{-*}|$ whose sign equals $sign_{i,j}$. Since the random mask r_c preserves the statistical distribution of the product, this replacement is statistically indistinguishable.

$\mathbf{Hyb_2}$: Replace $\widetilde{g_{i,j}}$ and $|\widehat{g_{i,j}}|$ with random values $\widetilde{g_{i,j}}^*$ and $|\widehat{g_{i,j}}|^*$, where $|\widehat{g_{i,j}}|^* = BC_{i,j} * \widetilde{g_{i,j}}^*$. Since the ratio $|\widehat{g_{i,j}}|^*/\widetilde{g_{i,j}}^* = BC_{i,j}$ remains unchanged, this replacement is computationally indistinguishable.

$\mathbf{Hyb_3}$: Replace the ciphertexts $[r_c|g_{i,j}^-|], [\widetilde{g_{i,j}}], [|\widehat{g_{i,j}}|], [G]$ with $[r_c|g_{i,j}^{-*}|], [\widetilde{g_{i,j}}^*], [|\widehat{g_{i,j}}|^*], [G^*]$. By the semantic security of CKKS, this replacement is computationally indistinguishable.

At this point, $Hyb_3 = IDEAL_{DS}$, thus the view of DS generated by the simulator S_{DS} in the ideal world is computationally indistinguishable from the view of DS in the real world.

Therefore, as the real and ideal views of both AS and DS are computationally indistinguishable, the protocol execution leaks no private information about users' gradients to either non-colluding server.

Theorem 3: BPFLH is resilient to Byzantine attacks when most users are benign.

Proof: In BPFLH, users with a confidence score less than a dynamic threshold are selected for aggregation. Specifically, we utilize the Bray-Curtis dissimilarity to compute the Bray-Curtis dissimilarity between the local gradients of users. The Bray-Curtis dissimilarity between the user U_i's local gradient g_i and the user U_j's local gradient g_j is defined as:

$$BC_{i,j} = \frac{|\widehat{g_{i,j}}|}{\widetilde{g_{i,j}}} = \frac{\sum_{k=1}^{d} ||g_i[k]| - |g_j[k]||}{\sum_{k=1}^{d} (|g_i[k]| + |g_j[k]|)}, \tag{21}$$

where $|g_i[k]|$ and $|g_j[k]|$ are the absolute values of the k-th element of gradient g_i and gradient g_j, respectively. Then, AS calculates the average Bray-Curtis dissimilarity between the local gradients of user U_i and all other users, which is utilized as the confidence score of user U_i:

$$C_i = \frac{\sum_{j=1, j \neq i}^{N} BC_{i,j}}{N-1}. \tag{22}$$

AS calculates the median $\overline{C}$ and standard deviation σ of all users' confidence scores C_i and sets a dynamic threshold $\theta = \overline{C} + m * \sigma$ to identify Byzantine users, where $m \in (0, 1)$ is a floating-point parameter used to adjust the sensitivity of the threshold. The Bray-Curtis dissimilarity is such that a value closer to 1 indicates a larger difference between local gradients, and a value closer to 0 indicates a smaller difference.

When a user's confidence score $C_i > \theta$, the user is classified as malicious. We assume that the majority of users are benign, i.e., $|U''|/N > 0.5$, where $|U''|$ is the number of benign users and N is the total number of users. When most of the users are benign, their confidence scores, which are based on the Bray-Curtis dissimilarity, will be relatively similar and concentrated around a certain value. This leads to a small standard deviation σ and a low median $\overline{C}$. As a result, both the median $\overline{C}$ and the standard deviation σ are small, which causes the dynamic threshold θ to be low. Even with a positive parameter m, the threshold remains small because it is influenced by the concentrated scores of the majority of benign users. Thus, the dynamic threshold θ, which is calculated within the range $[0, 1]$, will be much lower than 1 when the majority of users are benign.

All malicious users can collectively launch attacks to influence the global model. For a Byzantine user U_p to successfully carry out an attack, it must satisfy two conditions: (1) its confidence score C_p must be less than the dynamic threshold θ, i.e., $C_p \leq \theta$, and (2) there must be a significant difference between the gradients of the Byzantine user U_p and a benign user U_i, i.e., $|g_p| \gg |g_i|$, where $|g_p|$ and $|g_i|$ represent the absolute gradients of the Byzantine user U_p and the begin user U_i, respectively. This ensures that the gradient g_p of Byzantine user U_p can be aggregated and have a sufficiently negative impact on the global model. Since $|g_p| \gg |g_i|$, it implies that $|g_p[k]| \gg |g_i[k]|$, where $|g_p[k]||$ and $|g_i[k]|$ represent the absolute values of k-th element of Byzantine user U_p's gradient g_p and benign user U_i's gradient g_i, respectively.Thus, the confidence score C_p of Byzantine user U_p is

$$C_p = \frac{\sum_{j=1, j\neq p}^{N} BC_{p,j}}{N-1}.$$

Substituting in the Bray-Curtis dissimilarity formula:

$$C_p = \frac{1}{N-1}\sum_{j=1}^{N} \frac{\sum_{k=1}^{d} ||g_p[k]| - |g_j[k]||}{\sum_{k=1}^{d} ||g_p[k]| + |g_j[k]||}.$$

Given that $|g_p[k]| \gg |g_i[k]|$, we approximate $|g_p[k]| - |g_i[k]| \approx |g_p[k]|$ and $|g_p[k]| + |g_i[k]| \approx |g_p[k]|$. Further, we have

$$C_p \approx \frac{1}{N-1}\sum_{j=1}^{N} \frac{\sum_{k=1}^{d} ||g_p[k]||}{\sum_{k=1}^{d} ||g_p[k]||} = 1 > \theta,$$

since most users are benign. Thus, the Byzantine user U_p can be identified, and its gradient will not be aggregated. Hence, BPFLH is resilient to Byzantine attacks when most users are benign.

VII. Evaluation

A. Experimental settings

In this section, we present a detailed evaluation of the proposed BPFLH with respect to both accuracy, robustness and efficiency.

(1) Dataset:

· **MNIST:** The MNIST [40] dataset consists of grayscale images of handwritten digits ranging from 0 to 9, with each image represented as 28×28 pixel matrix. It contains 60,000 training samples and 10,000 testing samples.

· **Fashion-MNIST:** Fashion-MNIST [41] shares the same data format and size as MNIST but contains grayscale images of 10 categories of fashion items. It also includes 60,000 training samples and 10,000 samples.

· **CIFAR-10:** The CIFAR-10 [42] dataset comprises 60,000 color images of 32×32 pixels, categorized into 10 object or animal classes. It is split into 50,000 training samples and 10,000 testing samples.

In our experiments, the training sets of the above datasets are partitioned under both independent and identically distributed (IID) and non-independent and non-identically distributed (non-IID) settings.

(2) Model:

· **Convolutional Neural Network (CNN):** The CNN model employed in the experiments consists of two convolutional layers followed by two fully connected layers.

· **Multi-layer Perceptron (MLP):** The MLP model is composed of an input layer, one hidden layer, and an output layer.

(3) Attack Type:

· **Label Flipping:** In this attack, the labels of samples in a user's dataset are replaced with incorrect labels, while the data features of the samples remain unchanged. This leads to the model to learn valid feature representation but associate them with incorrect classes.

· **Model Poisoning:** In this attack, the parameters of a user's locally trained model are maliciously modified, thereby preventing the model from converging correctly.

(4) Experimental environment:

All experiments are conducted on a workstation equipped with an Intel(R) Xeon(R) Platinum 8255C 2.50GHz CPU and a GPU with 16GB of video memory, running Ubuntu 20.04. The implementation was carried out in Python and with CUDA acceleration.

(5) Experimental Parameters:

We set the learning rate of FL to $\mu = 0.01$, the local batch size to $B = 64$, the number of local training epochs to $LE = 5$, the total number of federated learning rounds to $FE = 100$, and the number of users to $N = 100$. For the CKKS polynomial, the degree is set to 8192 and the scaling factor is $s = 2^{40}$. Additionally, we employ the Dirichlet distribution to partition the dataset, simulating a non-IID distribution, with the concentration parameter of $\beta = 0.2$.

B. Experimental Results

(1) Accuracy evaluation:

In this experiment, we adopt FedAvg [2] as the baseline scheme, since it is one of the most representative aggregation algorithms in federated learning that does not incorporate any security mechanisms. We evaluate the performance of BPFLH compared with the classic FedAvg under different data partitioning methods, model architectures, and proportions of Byzantine users with distinct attack types. Specifically, experiments are conducted under both IID and non-IID settings, employing CNN and MLP models to assess the impact of label

TABLE II: Accuracy of BPFLH and FedAvg under IID data distribution against various attacks on different models and datasets

Scheme	Model	Percentage of Byzantine users (%)	CIFAR-10		Fashion-MNIST		MNIST	
Attacks			Label Flipping	Model poisoning	Label Flipping	Model poisoning	Label Flipping	Model poisoning
FedAvg	CNN	0	58.99	58.99	88.96	88.96	98.78	98.78
		10	55.34	21.69	87.77	57.57	97.99	61.57
		20	53.2	10.4	86.83	26.41	95.8	19.26
		30	51.6	9.32	79.52	12.09	66.74	17.95
	MLP	0	50.54	50.54	88.19	88.19	97.19	97.19
		10	49.3	19.17	87.83	50.11	96.84	50.15
		20	47.81	12.92	86.94	39.67	96.5	31.5
		30	44.69	9.76	83.63	11.86	94.32	14.6
BPFLH	CNN	0	58.99	58.99	88.96	88.96	98.78	98.78
		10	58.9	58.87	88.54	88.78	98.78	98.72
		20	58.81	58.93	88.91	88.37	98.73	98.68
		30	58.37	58.66	88.53	88.52	98.71	98.64
	MLP	0	50.54	50.54	88.19	88.19	97.19	97.19
		10	50.21	50.53	88.15	88.17	97.07	97.02
		20	50.18	50.37	88.12	88.14	96.93	96.98
		30	50.1	50.24	87.83	87.97	96.89	96.94

TABLE III: Accuracy of BPFLH and FedAvg under Non-IID data distribution against various attacks on different models and datasets

Scheme	Model	Percentage of Byzantine users (%)	CIFAR-10		Fashion-MNIST		MNIST	
Attacks			Label Flipping	Model poisoning	Label Flipping	Model poisoning	Label Flipping	Model poisoning
FedAvg	CNN	0	58.37	58.37	88.49	88.49	98.74	98.74
		10	56.52	19.05	87.36	49.6	98.23	64.3
		20	53.52	11.64	85.49	24.54	96.15	22.77
		30	49.55	9.44	77.29	18.21	63.63	18.36
	MLP	0	49.93	49.93	88	88	97.17	97.17
		10	48.31	14.75	87.44	50.02	96.96	48.34
		20	47.87	13.22	85.99	26.9	96.34	26.67
		30	45.13	10.07	82.59	14.82	94.01	18.8
BPFLH	CNN	0	58.37	58.37	88.49	88.49	98.74	98.74
		10	58.21	57.53	88.28	88.35	98.7	98.69
		20	58.13	57.33	88.19	88.24	98.66	98.6
		30	58.08	57.17	88.12	88.1	98.55	98.52
	MLP	0	49.93	49.93	88	88	97.17	97.17
		10	49.26	48.52	87.96	87.77	97.15	97.11
		20	48.97	48.21	87.2	87.59	96.93	97.15
		30	48.7	48.05	86.98	87.34	97.09	97.07

flipping and model poisoning attacks when the proportions of Byzantine users is $10\%, 20\%,$ and 30%.

1) IID Setting:

Tables II and III present the accuracy comparison between BPFLH and FedAvg under both IID and non-IID data distributions, respectively. As shown in Table II, under the IID data distribution, regardless of the models, datasets, or attack types, the performance of FedAvg degrades significantly as the proportion of Byzantine users increases. Moreover, model poisoning causes a more severe decline in accuracy than label flipping, as it directly manipulates the local model parameters, leading to drastic deviations in the global model. In contrast, label flipping mainly affects model training through incorrect label assignments, and the averaging aggregation mechanism of FedAvg mitigates its impact to some extent. By comparison, BPFLH demonstrates strong robustness against both label

TABLE IV: Accuracy comparison of different FL methods under various attacks on the MNIST dataset with IID data distribution (CNN model)

Percentage of Byzantine users	Attacks	FedAvg	Median	Trimmed-Median	Krum	FLTrust	PEFL	ShieldFL	PFLAD	CoS-HFL	BPFLH
0%	No attack	98.78	98.61	98.65	97.06	96.83	98.78	98.75	98.64	98.67	**98.82**
10%	Label Flipping	97.15	98.51	98.43	96.53	96.58	98.48	98.49	98.13	97.35	**98.78**
	Model Poisoning	61.57	98.54	98.58	96.82	96.16	98.19	98.59	98.61	98.01	**98.72**
20%	Label Flipping	96.60	98.36	98.07	96.78	96.42	98.39	93.49	97.49	98.25	**98.63**
	Model Poisoning	19.26	98.26	98.21	96.26	96.03	98.04	98.43	94.46	98.34	**98.68**
30%	Label Flipping	66.74	98.30	91.56	96.89	96.71	95.08	91.46	93.31	98.17	**98.79**
	Model Poisoning	17.95	98.06	97.53	96.99	96.82	98.03	98.47	93.07	98.31	**98.64**

TABLE V: Accuracy comparison of different FL methods under various attacks on the MNIST dataset with Non-IID data distribution (CNN model)

Percentage of Byzantine users	Attacks	FedAvg	Median	Trimmed-Median	Krum	FLTrust	PEFL	ShieldFL	PFLAD	CoS-HFL	BPFLH
0%	No attack	97.74	94.01	97.81	96.53	95.56	96.48	97.32	96.27	95.43	**97.85**
10%	Label Flipping	97.21	93.51	97.43	96.23	94.58	93.08	96.44	95.64	95.31	**97.75**
	Model Poisoning	64.30	93.64	97.58	86.32	94.16	84.19	96.51	95.61	95.07	**97.68**
20%	Label Flipping	96.15	93.39	95.07	95.68	94.42	90.39	95.89	94.68	95.23	**97.61**
	Model Poisoning	22.77	93.66	97.26	65.86	93.90	79.04	96.36	93.17	95.00	**97.42**
30%	Label Flipping	63.63	93.19	93.03	75.49	94.21	75.08	93.46	92.77	95.15	**97.20**
	Model Poisoning	18.36	93.91	97.05	55.78	93.45	68.03	95.85	91.52	95.31	**97.34**

flipping and model poisoning attacks across different datasets and models. Even when 30% of the users are Byzantine, BPFLH achieves accuracy comparable to that of the non-attacked FedAvg. This is because, in BPFLH, AS can accurately identify malicious gradients and aggregate only benign ones, thereby maintaining model performance on par with FedAvg. Furthermore, BPFLH achieves the robustness while simultaneously preserving user privacy and maintaining model accuracy.

2) Non-IID Setting:

From Table III, it can be observed that under the non-IID data distribution, the accuracy of FedAvg decreases more rapidly as the proportion of Byzantine users increases. This is primarily due to the combined effects of data heterogeneity and Byzantine attacks, which cause significant gradient divergence and exacerbate deviations in the global model. In particular, model poisoning continues to exert a stronger impact than label flipping, as directly manipulated parameters amplify inconsistencies among local updates. In contrast, BPFLH exhibits strong robustness under the non-IID setting. Across different datasets, models, Byzantine user ratios, and attack types, the accuracy of BPFLH remains close to that of the non-attacked FedAvg, even under the non-IID data distribution. This demonstrates that BPFLH can effectively distinguish malicious gradients from benign deviations induced by data heterogeneity, thereby maintaining high model accuracy.

(2) Robustness evaluation

To evaluate robustness, we set the proportion of Byzantine users to $10\%, 20\%$ and 30%. The compared schemes include the representative Byzantine-robust federated learning schemes Median [9], Trimmed Median [9], Krum [13], FLTrust [15], ShieldFL [16], PEFL [28], PFLAD [29] and CoS-HFL [17]. Specifically, Krum [13] selects the smallest distance gradients for aggregation. Median [9] replaces the aggregation operation with the median, while Trimmed Median [9] removes extreme gradients before aggregation. FLTrust [15] defends against Byzantine attacks by introducing a server model and calculating the similarity between user gradients and server model gradients. PEFL [28] identifies Byzantine users by measuring the linear correlation between user gradients. ShieldFL normalizes and compares cosine similarity between user gradients. PFLAD [29] extracts gradient features through a server-trained feature extraction and detects Byzantine users via cosine similarity between these features. CoS-HFL [17] defends against Byzantine attacks through a credit-based active poisoning resistance mechanism, combining multi-dimensional similarity and information entropy-based credibility evaluation and attacker elimination strategies.

Under the IID and non-IID setting, we trained the CNN and MLP models on the MNIST dataset and a CNN model on the CIFAR-10 dataset. Tables IV, V, VI, VII, VIII, and IX illustrate the model accuracy of different schemes under varying the attack types and Byzantine user proportion across different datasets and models.

Tables IV and V present the model accuracy of CNN trained on MNIST dataset under IID and non-IID distributions, respectively. As illustrated in Table IV, under IID conditions, BPFLH maintains an accuracy above 98.6% across label flipping and arbitrary model attacks, even with $10\%, 20\%$, and 30% Byzantine user proportions. This performance is almost identical to its performance in the absence of Byzantine users and demonstrates superior performance over all compared schemes. Under non-IID settings (Table V), BPFLH consistently achieves an accuracy of around 97.2% across all attack types and Byzantine user proportions. This slight reduction compared with the IID case primarily results from uneven data distribution, which increases gradient variance among users. Nevertheless, BPFLH still outperforms all compared schemes. In contrast, Krum and PEFL exhibit significant performance degradation as the proportion of Byzantine users increases, since both are designed under the IID assumption and struggle to adapt to heterogeneous data.

TABLE VI: Accuracy comparison of different FL methods under various attacks on the MNIST dataset with IID data distribution (MLP model)

Percentage of Byzantine users	Attacks	FedAvg	Median	Trimmed-Median	Krum	FLTrust	PEFL	ShieldFL	PFLAD	CoS-HFL	BPFLH
0%	No attack	97.27	97.10	97.14	96.82	97.21	97.13	97.11	96.32	97.08	**97.27**
10%	Label Flipping	97.09	95.50	95.90	96.33	97.05	96.91	96.35	96.08	96.26	**97.27**
	Arbitrary Model Attack	50.15	96.14	95.98	96.61	96.41	96.35	96.45	96.01	96.23	**97.22**
20%	Label Flipping	96.50	96.03	95.34	95.11	97.12	95.98	95.91	95.57	96.08	**97.18**
	Arbitrary Model Attack	31.50	95.36	95.49	93.72	97.19	96.60	96.45	95.18	96.54	**97.20**
30%	Label Flipping	74.32	95.19	95.44	95.17	96.55	96.12	94.08	95.28	97.00	**97.18**
	Arbitrary Model Attack	14.60	95.21	95.76	93.82	97.15	95.99	96.19	94.40	95.36	**97.17**

TABLE VII: Accuracy comparison of different FL methods under various attacks on the MNIST dataset with Non-IID data distribution (MLP model)

Percentage of Byzantine users	Attacks	FedAvg	Median	Trimmed-Median	Krum	FLTrust	PEFL	ShieldFL	PFLAD	CoS-HFL	BPFLH
0%	No attack	97.17	96.48	96.22	96.13	97.15	94.22	96.66	96.07	96.84	**97.15**
10%	Label Flipping	96.93	94.55	95.73	90.20	97.07	91.33	96.06	95.81	96.17	**97.11**
	Arbitrary Model Attack	63.28	93.53	95.64	71.53	97.09	70.51	95.93	95.15	96.35	**97.09**
20%	Label Flipping	93.93	93.70	94.68	82.35	96.95	83.58	95.46	94.27	95.21	**97.09**
	Arbitrary Model Attack	36.89	91.37	95.34	68.11	96.54	71.25	95.64	94.31	93.64	**97.12**
30%	Label Flipping	82.70	93.76	93.89	73.13	96.90	75.36	93.67	93.25	93.97	**97.09**
	Arbitrary Model Attack	18.80	91.63	94.30	60.57	97.01	56.55	95.97	93.14	92.77	**97.11**

TABLE VIII: Accuracy comparison of different FL methods under various attacks on the CIFAR-10 dataset with IID data distribution (CNN model)

Percentage of Byzantine users	Attacks	FedAvg	Median	Trimmed-Median	Krum	FLTrust	PEFL	ShieldFL	PFLAD	CoS-HFL	BPFLH
0%	No attack	61.99	58.36	59.19	57.66	59.13	59.26	58.17	59.53	60.50	**60.94**
10%	Label Flipping	56.49	57.76	59.16	57.44	58.61	58.48	57.66	59.04	59.45	**59.90**
	Arbitrary Model Attack	19.96	57.66	59.04	57.40	57.82	59.01	57.51	59.06	59.87	**60.87**
20%	Label Flipping	52.89	56.45	58.28	56.77	57.22	59.09	56.19	59.15	59.42	**60.78**
	Arbitrary Model Attack	10.40	57.92	58.13	55.14	57.70	58.94	57.35	59.09	58.56	**59.30**
30%	Label Flipping	47.65	57.85	58.84	56.71	57.88	59.15	55.32	59.45	58.64	**60.02**
	Arbitrary Model Attack	10.71	57.85	58.54	55.75	59.02	58.99	57.13	59.04	59.31	**59.26**

TABLE IX: Accuracy comparison of different FL methods under various attacks on the CIFAR-10 dataset with Non-IID data distribution (CNN model)

Percentage of Byzantine users	Attacks	FedAvg	Median	Trimmed-Median	Krum	FLTrust	PEFL	ShieldFL	PFLAD	CoS-HFL	BPFLH
0%	No attack	60.60	55.61	56.02	53.63	55.55	50.26	55.68	58.63	60.10	**60.67**
10%	Label Flipping	53.80	54.27	55.64	30.33	54.06	41.26	54.85	57.01	58.03	**58.49**
	Arbitrary Model Attack	19.05	53.76	55.64	26.39	54.04	42.05	53.52	57.53	56.32	**58.21**
20%	Label Flipping	50.55	53.73	55.33	28.11	53.45	45.26	53.90	56.33	57.61	**59.89**
	Arbitrary Model Attack	11.64	51.77	55.64	25.79	53.52	30.35	55.05	54.36	55.31	**59.30**
30%	Label Flipping	40.72	43.76	51.89	31.11	53.20	28.99	52.47	56.37	56.84	**59.58**
	Arbitrary Model Attack	11.31	53.63	54.30	23.82	51.72	41.95	54.03	52.67	55.01	**59.02**

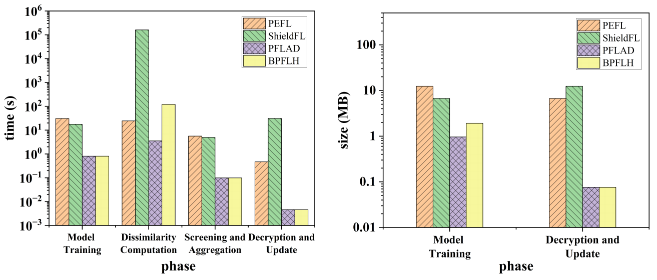
Tables VI and VII report the model accuracy of MLP trained on the MNIST dataset under IID and non-IID distributions, respectively. The results further confirm that even with a lighter model architecture, BPFLH maintains strong robustness against Byzantine attacks. As shown in Table VI, BPFLH achieves the highest accuracy across all attack types and Byzantine user proportions under IID conditions. As observed in Table VII, under the non-IID data distribution, BPFLH continues to demonstrate stable performance, maintaining an accuracy of approximately 97.1% and consistently outperforming the compared schemes across both label flipping and model poisoning attacks, as well as varying proportions of Byzantine users. The accuracy remains comparable to the IID setting, as BPFLH can accurately distinguish malicious gradients from benign deviations caused by data heterogeneity, thereby maintaining stable performance across different data distributions.

Tables VIII and IX present the model accuracy of CNN trained on CIFAR-10 with IID and non-IID distributions, respectively. As shown in Table VIII, under IID conditions, BPFLH preserves high accuracy against both label flipping and arbitrary model attacks, even when the proportion of Byzantine users increases to $10\%, 20\%$, and 30%. Furthermore, compared to the other schemes, BPFLH still achieves the highest accuracy under different Byzantine user proportions and attack types. Compared to Table IV, which shows the model accuracy of CNN trained on the MNIST dataset under IID data distribution, a noticeable drop in accuracy can be observed in Table VIII. This decline is mainly due to the higher complexity of the CIFAR-10 dataset, resulting in an accuracy of around 60%. From Table IX, it can be observed that under the non-IID distribution, most schemes exhibit significant accuracy drops. This is because, in non-IID data environment, benign users may produce gradients that deviate notably due to data heterogeneity. These schemes fail to effectively distinguish such deviating benign gradients from truly malicious gradients, misclassifying these deviating benign gradients as malicious, and thereby reducing the number of aggregated benign gradients. Even when adopting weights or scores for aggregation, these deviating benign gradients are assigned lower weights or scores, further weakening the model performance. In contrast, BPFLH detects Byzantine users by analyzing the discrepancies among user gradient elements, effectively mitigating the adverse impact of data heterogeneity on detection and keeping accuracy fluctuation within 2%.

Overall, by synthesizing the results across all datasets and model architectures, BPFLH consistently demonstrates superior robustness against Byzantine users, maintaining high accuracy across both IID and non-IID distributions.

(3) Efficiency Evaluation:

In this experiment, the proportions of Byzantine users is set at 0%, with the gradient dimension $d = 10000$. The computation and communication overheads of BPFLH are evaluated and compared with those of the several state-of-the-art Byzantine-robust privacy-preserving federated learning schemes, namely ShieldFL [16], PEFL [28], and PFLAD [29]. These schemes are selected for comparison because, similar to BPFLH, they employ homomorphic encryption to ensure data privacy while defending against Byzantine attacks. Specifically, PEFL and ShieldFL adopt Paillier homomorphic encryption, whereas PFLAD and BPFLH utilize CKKS homomorphic encryption for privacy protection. The experimental results are illustrated in Fig. 2.



(a) Computation overhead (b) Communication overhead

Fig. 2: The comparison of computation and communication overheads.

1) Computation overhead: In BPFLH, the computation overhead mainly arises from the model training, dissimilarity computation, screening and aggregation and decryption and update phases. As illustrated in Fig. 2(a), in the model training phase, PEFL incurs the highest computation overhead because it employs the Paillier homomorphic encryption, which is computationally complex and requires encrypting one ciphertext at a time. Similarly, ShieldFL also exhibits a comparably high computation overhead. In contrast, BPFLH and PFLAD achieve the lowest computation overhead in this phase, as both utilize the CKKS homomorphic encryption and encrypt data twice the size of the gradients, which is more efficient.

In the dissimilarity computation phase, all schemes compute gradient dissimilarity to identify potential Byzantine users. ShieldFL introduces the highest overhead in this phase since it performs multiple encryption and decryption operations. PFLAD exhibits the lowest computation overhead because its dissimilarity computation does not involve homomorphic operations and only requires performing simple encryption, decryption and local distance calculations. BPFLH requires executing homomorphic addition and multiplication operations, which result in slightly higher computation overhead compared to PFLAD. However, this additional overhead is a reasonable trade-off for achieving higher accuracy in precisely distinguishing malicious gradients from benign deviations caused by data heterogeneity.

In the screening and aggregation phase, PFLAD, BPFLH, and PEFL all detect and remove malicious gradients before aggregation, while ShieldFL aggregates all gradients with different weights. The computation overhead of PEFL and ShieldFL is significantly higher than that of PFLAD and BPFLH, owing to the lower efficiency of the Paillier scheme compared with CKKS.

In the decryption and update phase, all schemes perform decryption and model updates. Because CKKS operations are more efficient than those of Paillier, PFLAD and BPFLH incur lower overhead than PEFL and ShieldFL. It is worth noting that ShieldFL, which uses an extended Paillier encryption scheme, involves more complex decryption, resulting in the highest computation cost in this phase.

2) Communication overhead: The communication cost of BPFLH is mainly incurred during the model training and the decryption and update phases. In the model training phase, all these schemes involve local training and the encryption and uploading of gradients. Hence, the primary source of communication overhead comes from transmitting the encrypted gradients. Fig. 2(b) shows that in the model training phase, PEFL incurs the highest communication overhead since it needs to download the encrypted global model first, decrypts it for local training and gradient computation, and then encrypts and uploads the gradients. Consequently, the communication overhead of PEFL is approximately twice the ciphertext size of the gradients. The communication overhead of ShieldFL is slightly lower than that of PEFL, as ShieldFL performs a preliminary screening of the gradients and uploads only those with higher contributions. The communication overhead of PFLAD is the lowest. This advantage stems from its use of CKKS homomorphic encryption, which supports vectorized encryption, allowing multiple vectors to be encrypted into a single ciphertext. BPFLH incurs slightly higher communication overhead than PFLAD since BPFLH also transmits the encrypted absolute gradients in addition to the encrypted gradients. This additional overhead is a justified trade-off for achieving enhanced robustness, particularly in accurately identifying Byzantine users under non-IID conditions, while also providing privacy protection.

In the decryption and update phase, ShieldFL shows the highest communication overhead, as it transmits not only the encrypted aggregated gradient but also additional partial decryption results, leading to extra communication overhead.

By contrast, PFLAD and BPFLH achieve lower communication overhead than PEFL and ShieldFL, because in PFLAD and BPFLH, the aggregated gradients are decrypted on the server side before being returned to users, whereas PEFL and ShieldFL transimit the encrypted aggregated gradients to users before performing decryption.

C. Discussion and limitation

Although BPFLH uses CKKS for privacy-preserving calculations, it is not limited to this encryption scheme. In principle, other homomorphic encryption methods, such as Paillier, can also be applied, as long as floating-point numbers are converted into integers and the calculations are performed element-wise. CKKS is chosen primarily because it natively supports direct floating-point number computations, which are essential for Bray-Curtis dissimilarity calculation. Moreover, CKKS allows efficient vectorized operations, significantly improving computational performance.

While the proposed BPFLH demonstrates strong robustness and privacy-preserving performance on standard benchmarks such as MNIST [40], Fashion-MNIST [41], and CIFAR-10 [42], several limitations remain. First, the current experimental evaluation is constrained by computational resources, particularly due to the high cost of homomorphic encryption operations in large-scale federated settings. Consequently, more complex or domain-specific datasets were not incorporated in this study. In future work, we plan to extend BPFLH to large-scale datasets such as FEMNIST, which involve thousands of heterogeneous clients, as well as to privacy-sensitive domains like medical imaging (e.g., COVID-Xray), where data heterogeneity and security requirements are more pronounced.

VIII. Conclusion

In this paper, we propose BPFLH, a novel Byzantine robustness and privacy-preserving federated learning scheme for heterogeneous data. We employ Bray–Curtis dissimilarity to accurately identify Byzantine users and use CKKS homomorphic encryption to protect privacy, enhancing the robustness against Byzantine attacks in non-IID data environments while preserving user privacy. Comprehensive experiments under both IID and non-IID settings confirm that BPFLH achieves consistently high accuracy across diverse datasets, model architectures, and Byzantine attack types (label flipping and model poisoning), even with varying proportions of Byzantine users, while ensuring privacy protection. In the future, we will explore new encryption methods compatible with the Bray-Curtis dissimilarity calculation to optimize efficiency.

References

[1] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[3] G. Xu, H. Li, Y. Zhang, S. Xu, J. Ning, and R. H. Deng, "Privacy-preserving federated deep learning with irregular users," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 2, pp. 1364–1381, 2020.

[4] C. Xu, Y. Jia, L. Zhu, C. Zhang, G. Jin, and K. Sharif, "Tdfl: Truth discovery based byzantine robust federated learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 12, pp. 4835–4848, 2022.

[5] Y. Miao, R. Xie, X. Li, Z. Liu, K.-K. R. Choo, and R. H. Deng, "Efficient and secure federated learning against backdoor attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 5, pp. 4619–4636, 2024.

[6] Z. Zhang, J. Li, S. Yu, and C. Makaya, "Safelearning: Secure aggregation in federated learning with backdoor detectability," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 3289–3304, 2023.

[7] X. Lyu, Y. Han, W. Wang, J. Liu, B. Wang, K. Chen, Y. Li, J. Liu, and X. Zhang, "Coba: Collusive backdoor attacks with optimized trigger to federated learning," *IEEE Transactions on Dependable and Secure Computing*, 2024.

[8] X. Sheng, Z. Yang, and W. Bao, "Fairguard: A fairness attack and defense framework in federated learning," *IEEE Transactions on Dependable and Secure Computing*, 2024.

[9] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *International conference on machine learning*. Pmlr, 2018, pp. 5650–5659.

[10] Y. Miao, Z. Liu, H. Li, K.-K. R. Choo, and R. H. Deng, "Privacy-preserving byzantine-robust federated learning via blockchain systems," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 2848–2861, 2022.

[11] L. Yang, Y. Miao, Z. Liu, Z. Liu, X. Li, D. Kuang, H. Li, and R. H. Deng, "Enhanced model poisoning attack and multi-strategy defense in federated learning," *IEEE Transactions on Information Forensics and Security*, 2025.

[12] Z. Zhang, L. Wu, C. Ma, J. Li, J. Wang, Q. Wang, and S. Yu, "Lsfl: A lightweight and secure federated learning scheme for edge computing," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 365–379, 2022.

[13] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *Advances in neural information processing systems*, vol. 30, 2017.

[14] C. Fang, Z. Yang, and W. U. Bajwa, "Bridge: Byzantine-resilient decentralized gradient descent," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 8, pp. 610–626, 2022.

[15] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "Fltrust: Byzantine-robust federated learning via trust bootstrapping." in *NDSS*, vol. 21, no. 2, 2021, pp. 1–15.

[16] Z. Ma, J. Ma, Y. Miao, Y. Li, and R. H. Deng, "Shieldfl: Mitigating model poisoning attacks in privacy-preserving

federated learning," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 1639–1654, 2022.

[17] Z. Zhou, J. Zhao, S. Yang, H. Li, T. Ma, and C. Xu, "Community-oriented duplex privacy amplification and active poisoning resistance for heterogeneous federated learning," *IEEE Transactions on Dependable and Secure Computing*, 2025.

[18] Q. Dong, S. Yang, Z. Dai, Y. Gao, S. Wang, Y. Cao, A. Fu, and W. Susilo, "Carefl: Contribution guided byzantine-robust federated learning," *IEEE Transactions on Information Forensics and Security*, 2024.

[19] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *2019 IEEE symposium on security and privacy (SP)*. IEEE, 2019, pp. 739–753.

[20] C. Ma, J. Li, M. Ding, H. H. Yang, F. Shu, T. Q. Quek, and H. V. Poor, "On safeguarding privacy and security in the framework of federated learning," *IEEE network*, vol. 34, no. 4, pp. 242–248, 2020.

[21] Y. Miao, Z. Liu, X. Li, M. Li, H. Li, K.-K. R. Choo, and R. H. Deng, "Robust asynchronous federated learning with time-weighted and stale model aggregation," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 4, pp. 2361–2375, 2023.

[22] Y. Cai, W. Ding, Y. Xiao, Z. Yan, X. Liu, and Z. Wan, "Secfed: A secure and efficient federated learning based on multi-key homomorphic encryption," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 4, pp. 3817–3833, 2023.

[23] V. Vo, M. Ma, G. Bai, R. Ko, and S. Neplal, "Practical poisoning attacks with limited byzantine clients in clustered federated learning," in *2025 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2025, pp. 1751–1769.

[24] X. Gui, G. Yu, J. Wang, Z. Yan, W. Wang, C. Domeniconi, and L. Cui, "Sophon: Byzantine-robust federated learning via dual trust mechanism," *IEEE Transactions on Dependable and Secure Computing*, 2025.

[25] Y. Mao, Z. Ye, X. Yuan, and S. Zhong, "Secure model aggregation against poisoning attacks for cross-silo federated learning with robustness and fairness," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 6321–6336, 2024.

[26] H. Yan, C. Zheng, Q. Chen, X. Li, B. Wang, H. Li, and X. Lin, "A proactive defense against model poisoning attacks in federated learning," *IEEE Transactions on Dependable and Secure Computing*, 2025.

[27] J. So, B. Güler, and A. S. Avestimehr, "Byzantine-resilient secure federated learning," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 7, pp. 2168–2181, 2020.

[28] X. Liu, H. Li, G. Xu, Z. Chen, X. Huang, and R. Lu, "Privacy-enhanced federated learning against poisoning adversaries," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 4574–4588, 2021.

[29] P. Tang, X. Zhu, W. Qiu, Z. Huang, Z. Mu, and S. Li, "Flad: Byzantine-robust federated learning based on gradient feature anomaly detection," *IEEE Transactions on Dependable and Secure Computing*, 2025.

[30] Y. Chen, W. Tan, Y. Zhong, Y. Kang, A. Yang, and J. Weng, "Byzantine-robust and privacy-preserving federated learning with irregular participants," *IEEE Internet of Things Journal*, 2024.

[31] M. Xhemrishi, J. Östman, A. Wachter-Zeh, and A. G. i Amat, "Fedgt: Identification of malicious clients in federated learning with secure aggregation," *IEEE Transactions on Information Forensics and Security*, 2025.

[32] R. Xue, K. Xue, B. Zhu, X. Luo, T. Zhang, Q. Sun, and J. Lu, "Differentially private federated learning with an adaptive noise mechanism," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 74–87, 2023.

[33] X. Tang, C. Guo, K.-K. R. Choo, and Y. Liu, "An efficient and dynamic privacy-preserving federated learning system for edge computing," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 207–220, 2023.

[34] H. Zeng, J. Lou, K. Li, C. Wu, G. Xue, Y. Luo, F. Cheng, W. Zhao, and J. Li, "Esfl: Accelerating poisonous model detection in privacy-preserving federated learning," *IEEE Transactions on Dependable and Secure Computing*, 2025.

[35] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Advances in cryptology–ASIACRYPT 2017: 23rd international conference on the theory and applications of cryptology and information security, Hong kong, China, December 3-7, 2017, proceedings, part i 23*. Springer, 2017, pp. 409–437.

[36] J. R. Bray and J. T. Curtis, "An ordination of the upland forest communities of southern wisconsin," *Ecological monographs*, vol. 27, no. 4, pp. 326–349, 1957.

[37] X. Li, X. Yang, Z. Zhou, and R. Lu, "Efficiently achieving privacy preservation and poisoning attack resistance in federated learning," *IEEE Transactions on Information Forensics and Security*, 2024.

[38] Z. Liu, H. Ye, Y. Jiang, J. Shen, J. Guo, I. Tjuawinata, and K.-Y. Lam, "Privacy-preserving federated unlearning with certified client removal," *IEEE Transactions on Information Forensics and Security*, 2025.

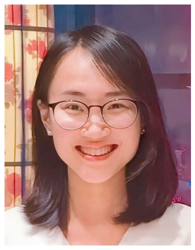
[39] Y. Miao, X. Yan, X. Li, S. Xu, X. Liu, H. Li, and R. H. Deng, "Rfed: Robustness-enhanced privacy-preserving federated learning against poisoning attack," *IEEE Transactions on Information Forensics and Security*, 2024.

[40] L. Yann, "Mnist handwritten digit database," *ATT Labs.*, 2010.

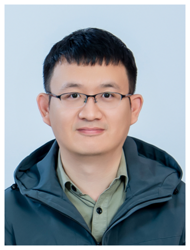
[41] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[42] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

Guofu Zhu received the Bachelor of Engineering degree in Computer Science and Technology from the School of Computer Science and Technology, Shandong Jianzhu University, China, in 2022. He is currently pursuing his Master's degree in Cybersecurity at the School of Computer Science and Technology, Qingdao University. His research interests include privacy protection and federated learning.



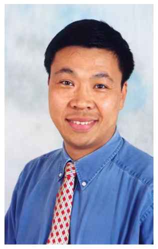
Wenting Shen received Ph. D. degree in School of Mathematics from Shandong University, in 2020. She is currently an associate professor of the College of Computer Science and Technology at Qingdao University. She has published several research papers in refereed international journals, including IEEE TIFS, IEEE TDSC, IEEE TSC, etc. Her research interests include cloud computing security, privacy computing, and big data security.



Zhiquan Liu received the B.S. degree from the School of Science, Xidian University, Xi'an, China, in 2012, and the Ph.D. degree from the School of Computer Science and Technology, Xidian University, Xi'an, China, in 2017.

He is currently a full professor, doctoral supervisor, and deputy dean with the College of Cyber Security, Jinan University, Guangzhou, China. His current research focuses on security, trust, privacy, and intelligence in vehicular networks and UAV networks. He currently serves as the area editor or associate editor of multiple SCI-index journals, such as IEEE TIFS, IEEE TDSC, IEEE TII, IEEE TVT, IEEE IOTJ, IEEE Network, Information Fusion, etc. His homepage is https://www.zqliu.com.

Jing Qin received the BS degree from Information Engineering University, Zhenzhou, China, in 1982, and the PhD degree from the School of Mathematics, Shandong University, China, in 2004. She is a professor with the School of Mathematics, Shandong University China. Her research interests include information security, design and analysis of security about cryptologic protocols. She has coauthored two books and has published more than 30 professional research papers. She is a senior member of Chinese association for Cryptologic Research (CACR) as well as China Computer Federation (CCF).



Jixin Ma is a Full Professor of Computer Science (Artificial Intelligence) and the Director of PhD/Postgraduate Research Programme in the School of Computing and Mathematical Sciences at University of Greenwich, U.K. He has been the Director of the Centre for Computer and Computational Science and the Lead of Artificial Intelligence Research Group. Professor Ma is also a Visiting Professor of Beijing Normal University, Hainan University, Anhui University, Zhengzhou Light Industrial University and Macau City University. Professor Ma obtained his BSc and MSc of Mathematics in 1982 and 1988, respectively, and PhD of Computer Sciences in 1994. His main research areas include Artificial Intelligence, Data Science, and Information Systems, with special interests in Temporal Logic, Information Security, Machine Learning, Case-Based Reasoning and Pattern Recognition. Professor Ma has been a member British Computer Society, American Association of Artificial Intelligence, ACIS/IEEE, World Scientific and Engineering Society, and Special Group of Artificial Intelligence of BCS. He has also been the Editor of several international journals and international conference proceedings, Conference/Program Chair, and Invited Keynote Speakers of many international conferences. Professor Ma has published more than 200 research papers in peer-reviewed international journals and conferences.